Player Function keys 11

Fx key 12

Pitch/Tempo key 13

Player Function keys 11

Mode keys 16

20

Mode keys 16

Volume keys 14

10

Joystick 15

Save/Edit key 17

ON/OFF key 18

FIG. 1

▶	Play
■	Stop
▶▶	Forward
◀◀	Reverse
●	Record

Player Function keys

FIG. 2

♫	e.DJ
(((A)))	V.Radio
◉	Songs
▤	Samples
⚙	System

Mode/Direct Access keys

FIG. 3

FIG. 4
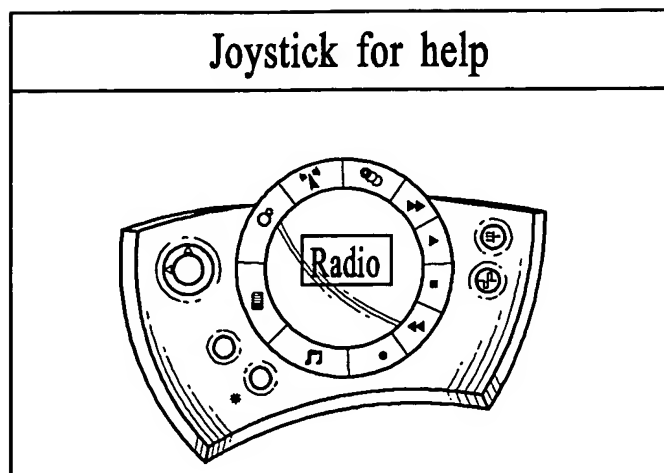
Home Screen

FIG. 5
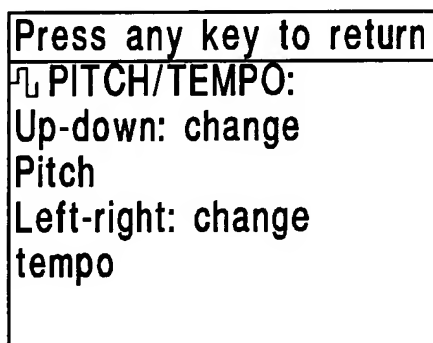
Help Screen

FIG. 6

e.DJ Style Selection Screen

SONG INTRO

Riff lane
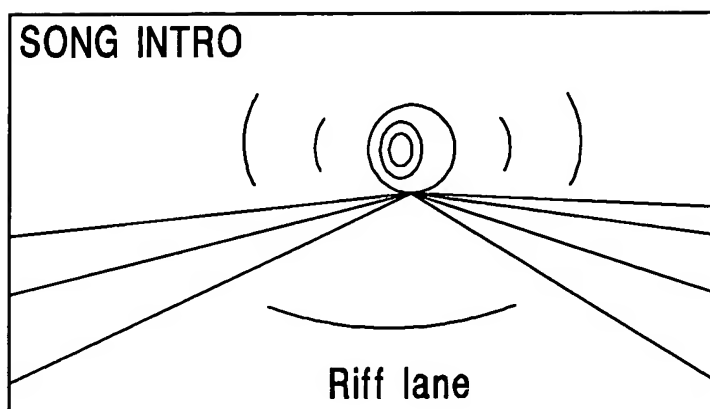
e.DJ I-Way Screen

FIG. 7

RIFF #1
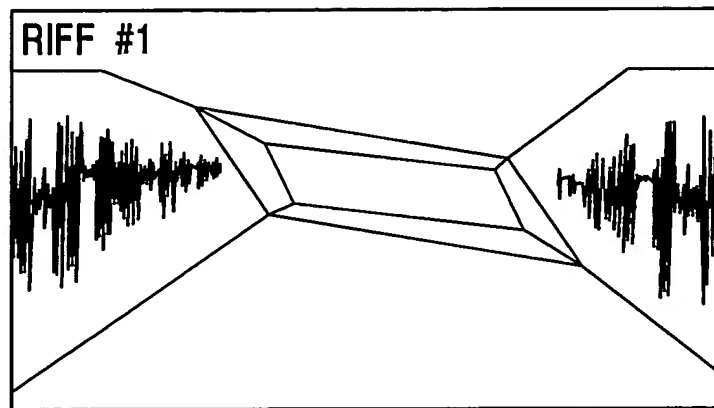
e.DJ Underground Screen

FIG. 8

FIG. 9

e.Songs

Fragile
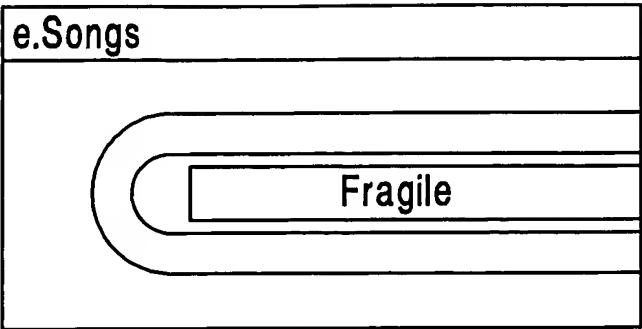
Play Song Screen

FIG. 10

v.Radio   96.0 MHz

Radio01

Play Radio Screen

FIG. 11

New SONGLIST001

1   .JINGLE
2   ALLNIGHT
3   FRAGILE
4   GROOVE
5   END LIST
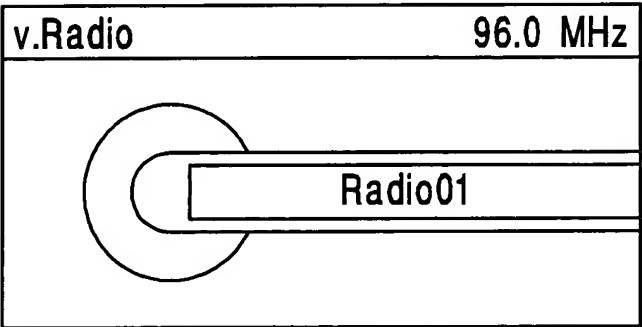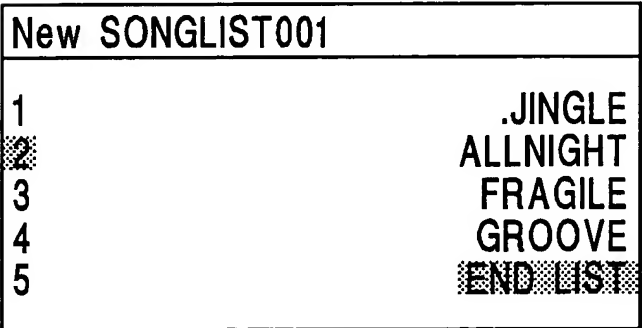
List Edit Screen

FIG. 12

Configuration

AUTOPLAY   OFF
POWER OFF   DISABLED
AUTOREPEAT   40 ms
EQ PRESETS   DEFAULT
STATION SEARCH   AUTO
REC FORMAT   PCM
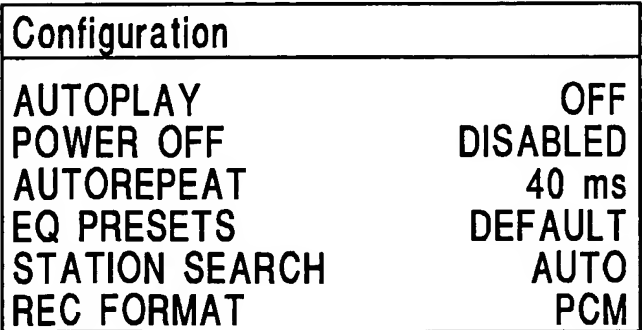
Configuration Screen
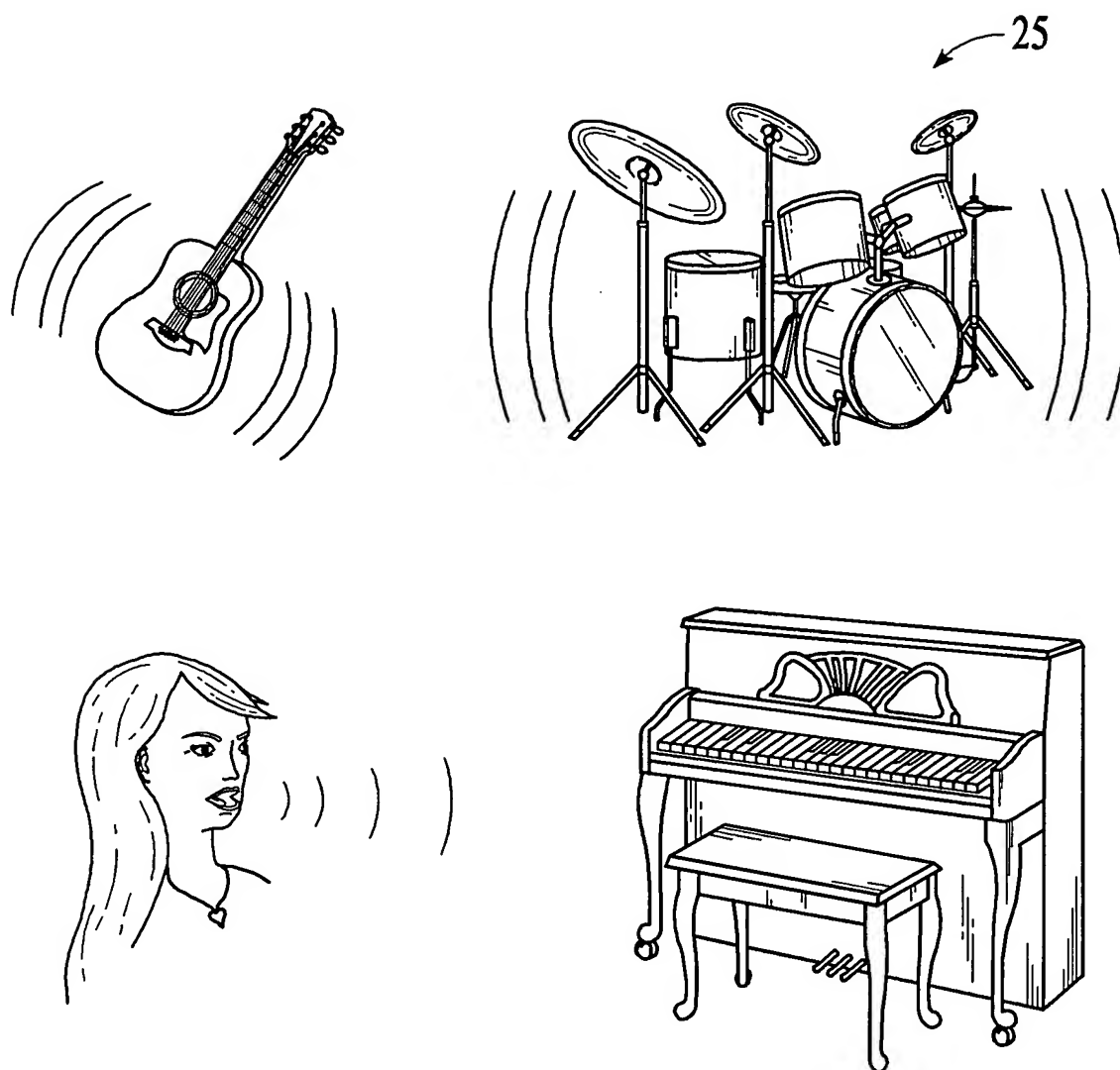
Alternative User Interface for I-Way Mode

FIG. 13

Parameter	Values	Description
AutoPlay	On/Off	If AutoPlay is On, the MadPlayer automatically starts playing the first Play list contained on a SmartMedia card when inserted.
Power Off	Disabled, 1mn to 60mn in steps of 1mn.	Auto power off delay. The MadPlayer will power off automatically after this delay if no user action is detected.
AutoRepeat	40ms to 600ms in steps of 20ms	Keyboard auto-repeat delay in milliseconds. Delay before repeating the corresponding action when a key is pressed continuously.
EQ Preset	Factory Woof Hitek Flat User	Presets for 4-band equalizer. Factory, Woof, HiTek and Flat are factory presets and fixed. User preset can be configured by the User via the System-Equalizer menu.
Mic State	On/Off	Microphone input is On or Off.
Mic Volume	0 to 31	Microphone volume.
Echo Level	0 to 127	Level of echo applied to microphone input
Echo Time	0 to 127	Microphone echo delay. 0 shortest, 127 longest.
Echo Feedbk	0 to 31	Echo feedback: 0 minimum feedback, 127 maximum feedback.
Rec Format	PCM HQFADPCM	Format used to store recorded samples: PCM: PCM, 16bits mono, 19.31kHz HQFADPCM: High Quality ADPCM
Language	English Francais Espanol	Language used for the menus.
Sort Files	By Name By Type	Criterion used to sort files when displaying a list: by name (alphabetically) or by type (songs, samples, lists...).
Sort Presets	By Name By Freq	Criterion used to sort radio presets: by name (alphabetically) or by frequency.
Product	String	Read Only. Hardware version
Release	String	Read Only. Firmware version

Configuration Parameters

FIG. 14

Song | P_0 | P_1 | P_2 | P_3 | P_1 | P_2 | P_4 | | Ending |

Part | SP_0 | SP_1 | SP_2 | SP_3 | SP_4 | SP_5 | SP_6 | | SP_n |

SubPart | SEQ_0 | SEQ_1 | SEQ_2 | SEQ_3 | SEQ_4 | SEQ_5 | SEQ_6 | | SEQ_n |

Sequence
RP_{ax}	RP_{ay}
RP_{bx}	RP_{by}
RP_{cx}	RP_{cy}
RP_{dx}	RP_{dy}

Real Pattern | RP_x | RP_y |

Virtual Pattern | B_1 | B_2 | B_3 | B_4 | B_5 | | B_n |

Block
| $Block_d$ |
| $Block_{fx}$ |

Sub Blocks
| SB_x |
| SB_y |
| SB_z |
| SB_n |

Song Structure

FIG. 15

```
                    ┌──────────────────┐
                    │    Style and     │
                    │    Instrument    │
                    └──────────────────┘
                             │
                             ▼          1. Apply Block Rules
                    ┌──────────────────┐
                    │ Virtual Pattern Sub │
                    │      Blocks      │
                    └──────────────────┘
                             │
                             ▼          2. Apply rhythmic rules to combine into series
                    ┌──────────────────┐
                    │     Virtual      │
                    │     Pattern      │
                    └──────────────────┘
                             │
                             ▼          3. Apply musical rules to generate basic music
                    ┌──────────────────┐
                    │   Non-Chorded    │
                    │  Pattern (NCP)   │
                    └──────────────────┘
                             │
                             ▼          4. Apply Tonic
                    ┌──────────────────┐
                    │     NCP with     │
                    │      Tonic       │
                    └──────────────────┘
                             │
                             ▼          5. Apply Mode
                    ┌──────────────────┐
                    │ NCP with Tonic & │
                    │      Mode        │
                    └──────────────────┘
                             │
                             ▼          6. Apply Key
                    ┌──────────────────┐
                    │   Real Pattern   │
                    └──────────────────┘
                             │
                             ▼          7. Apply any global pitch adjustment in real
```

General Musical Generation Flow

FIG. 16

Hexadecimal Value	Internal Nomenclature	Potential Values
40	Base Note	**C, E, G, B**
41	Magic Note 1	**+1, −1, +2, −2**
42	Magic Note 0	**+1, −1, +2, −2, 0**
43	High Note	**+7**
44	Last Note	**C, G**
45	One Before Last Note	**E, G, B**
46	ALC Controller • Harmonic Note • Fixed Note	**0, +2, +4, +6, -3, -5, -7** **any**

Examples of Virtual Notes/Controllers

FIG. 17

Treble Instrument, such as a violin, with an optimum range that is relatively high in pitch.

Midrange Instrument, such as a guitar, with an optimum range that is relatively medium in pitch.

Bass Instrument, such as a bass guitar, with an optimum range that is relatively low in pitch.
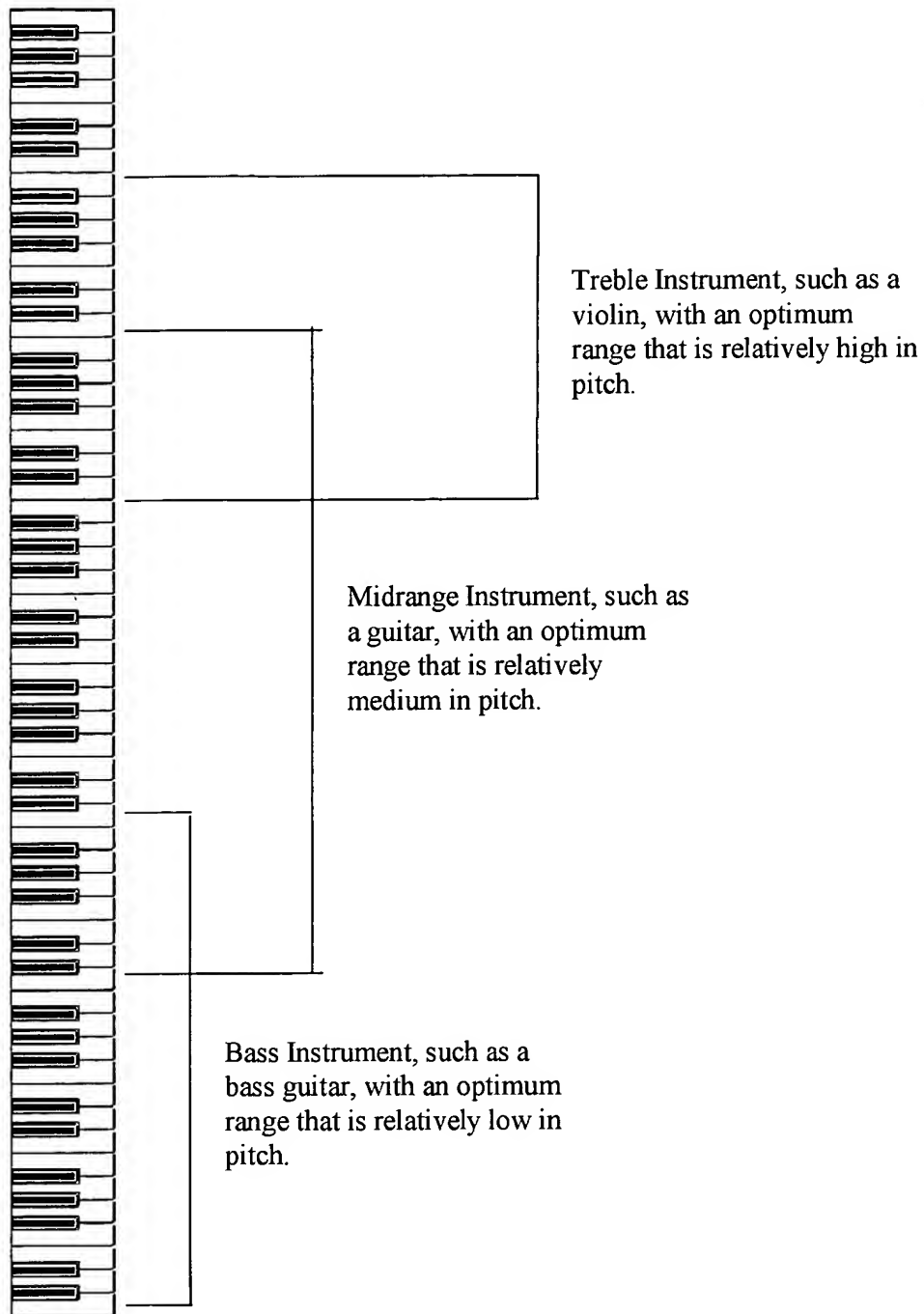
Example of Tessitura

FIG. 18
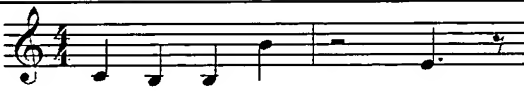
Chord	Key			
	A	C	D	G
Offset	-3	0	+2	+8

FIG. 19
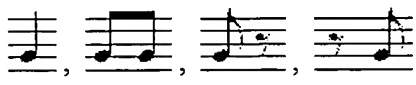
Mode Type	Individual Notes											
All Notes	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
Natural	C	C	D	D	E	F	F	G	G	A	A	B
Lydian Descending	C	C	D	D	E	E	F#	G	G	A	A	B
Lydian Ascending	C	D	D	E	E	F#	F#	G	A	A	A	B

FIG. 20

	Musical Notation	Software Notation (QN=30)
Virtual Pattern Sub-Blocks		C4 = Base Note F#4 = Magic Note Type 1 D4 = Magic Note Type 0 C#4 = High Note C4 = Base Note
Virtual Pattern (VP)		00 91 30 70 1e 81 30 00 91 36 64 1e 81 36 00 91 32 7f 1e 81 32 00 91 31 72 1e 81 31 3C 91 30 64 2d 81 30
Non-Chorded Pattern (NCP)		00 91 34 70 1e 81 34 00 91 32 64 1e 81 32 00 91 32 7f 1e 81 32 00 91 3e 72 1e 81 3e 3C 91 37 64 2d 81 37
NCP with Tonic (PwT)		00 91 31 70 1e 81 31 00 91 2f 64 1e 81 2f 00 91 2f 7f 1e 81 2f 00 91 3b 72 1e 81 3b 3C 91 34 64 2d 81 34
PwT with Mode (PwTM)		00 91 30 70 1e 81 30 00 91 2f 64 1e 81 2f 00 91 2f 7f 1e 81 2f 00 91 3b 72 1e 81 3b 3C 91 34 64 2d 81 34
Real Pattern (RP)		00 91 32 70 1e 81 32 00 91 31 64 1e 81 31 00 91 31 7f 1e 81 31 00 91 3d 72 1e 81 3d 3C 91 36 64 2d 81 36

Example of VP-to-RP Flow

FIG. 21

Rhythmic Blocks/Sub-Blocks	Conditions
Relative Rhythmic Density	All variations, given: • eighth note is smallest unit • length of 1 quarter note • all full rests are indicated separately as 'empty'
	All variations, given: • eighth note is smallest unit • length of 2 quarter notes • does not include 1 quarter note variations above

Rhythmic Variations based on Duration

FIG. 22

More Magic Notes

Pitch

More Base and High Notes

Time
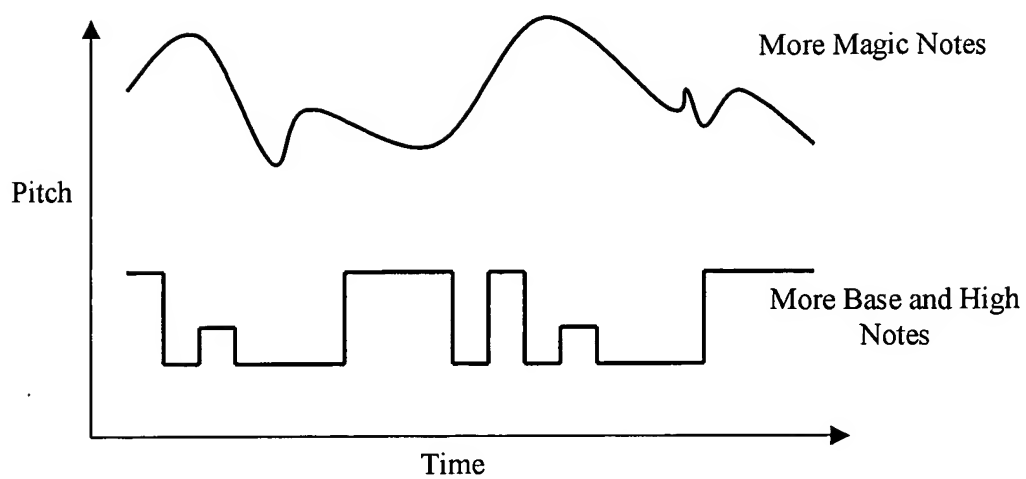
Relative Mobility of Note Pitch

FIG. 23

Patt_Info [Shift] [Num_Types * Num_Sub_Drums] {Block_Ind, FX_No$_1$, Combi_No}

Comb_Index_List [Styles * Num_Types] {Style$_i$_Type$_j$_Combi_Index}

{SubStyle_Mask$_k$, Combi_Index$_k$, Group_Index$_k$}

Style$_i$_Type$_j$_Combi {Block_Size$_{pq}$}

Style _Group {BANK $_k$PCk, {P $_k$} GS $_k$

" " "

" " "

}
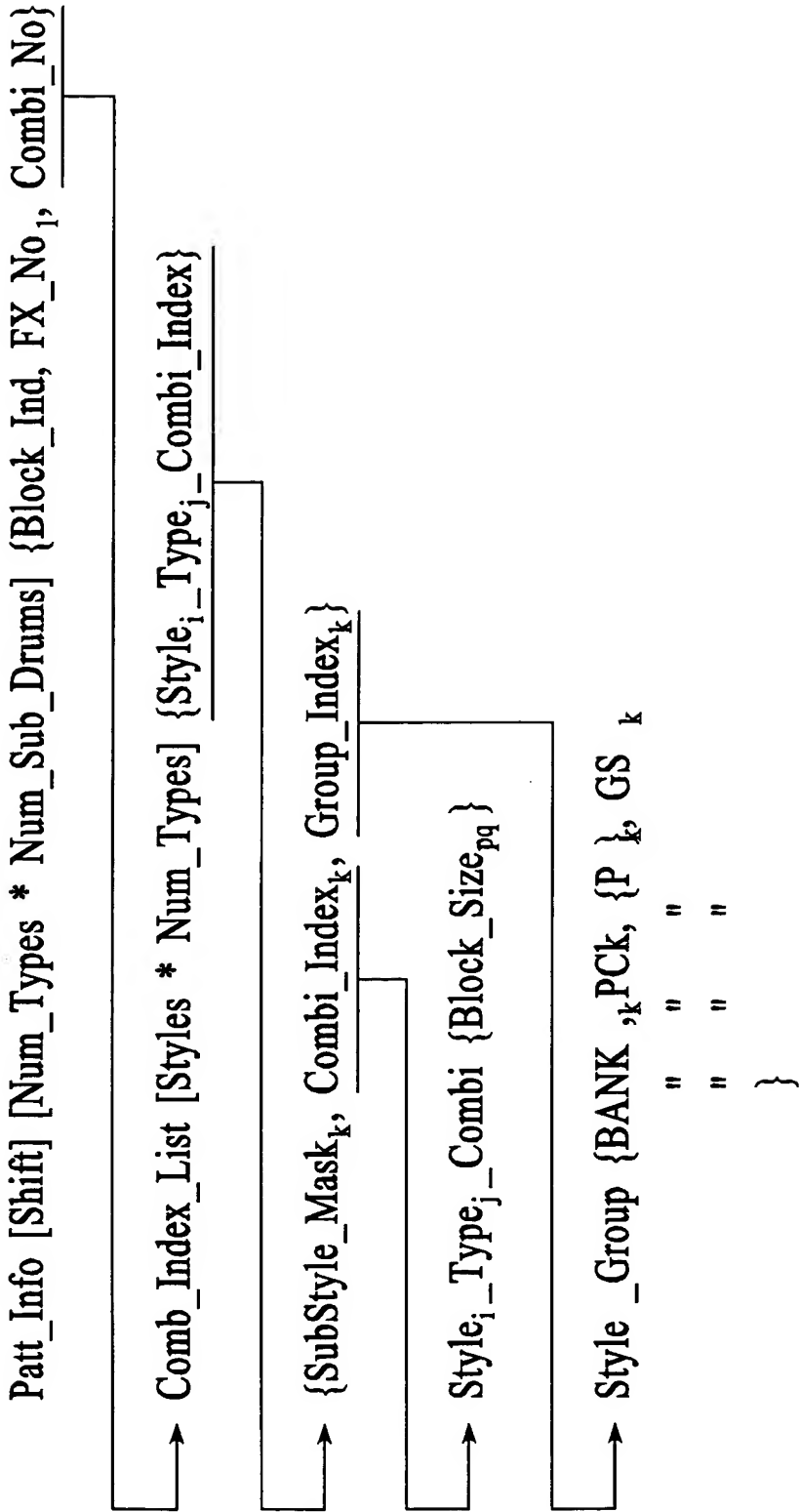
Pattern Structure Creation Example

FIG. 24

Block_List [Style * Num_Types * Sum_Sub_Drums] {Style, Type$_g$, Block_Index}

{Width$_k$, SubStyle_Mask, Group$_k$, Start_Pointer$_k$}

{Virtual_Block_Data}
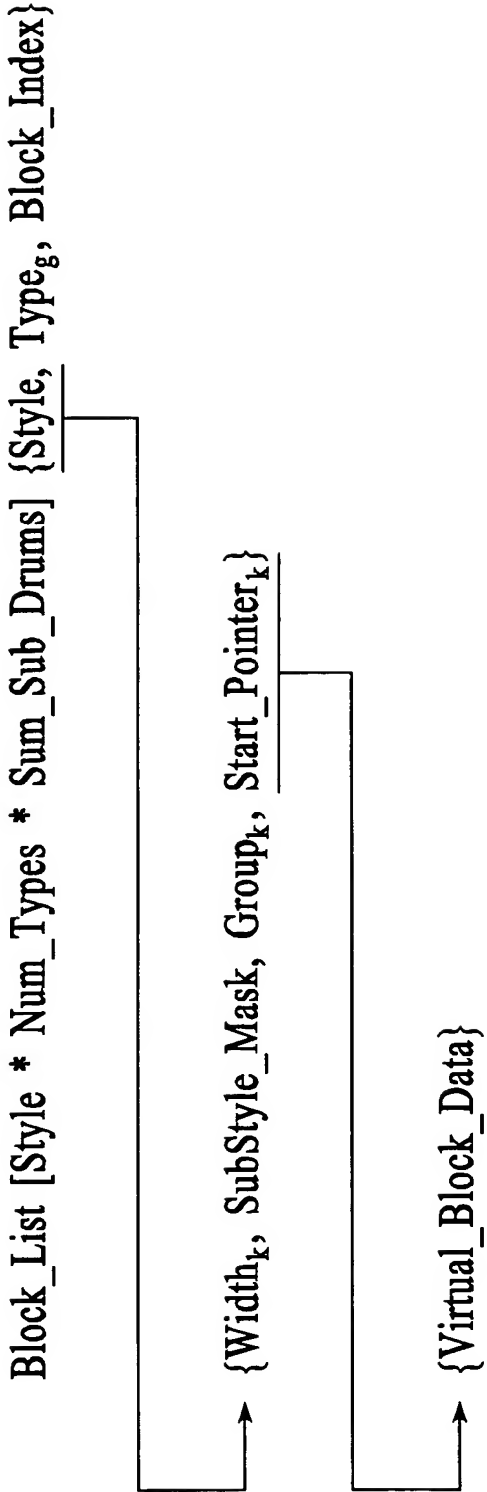
Block Structure Creation Example

FIG. 25

Drum Seed (DS)
A

B

DS_0

DS_1

DS_2

DS_3

Drum Part
Generation
Algorithm

C

Drum Part D
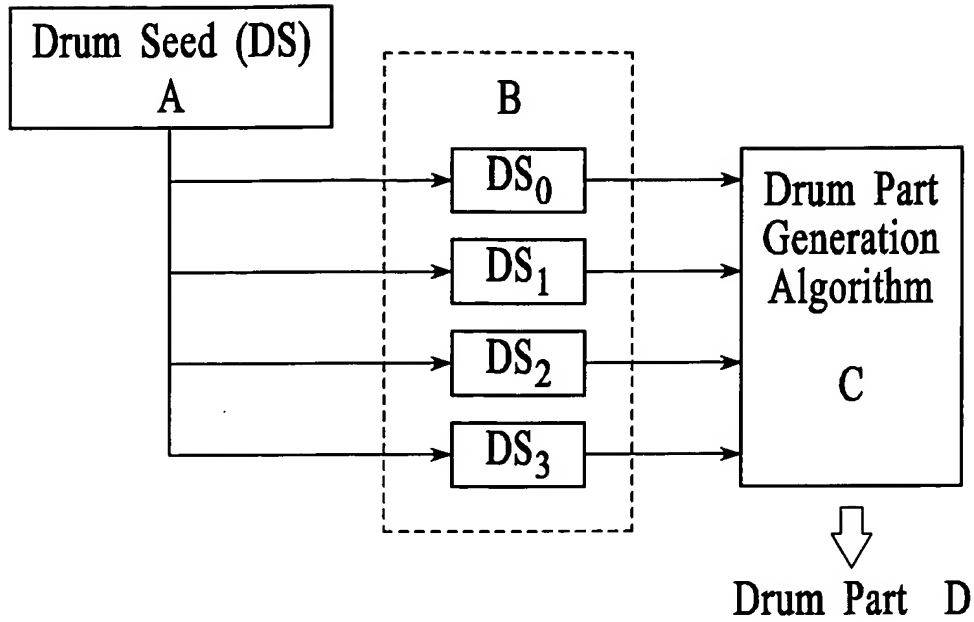
Pseudo-Random Number Implementation 1

FIG. 26

Bass Seed (BS)
A'

B'

BS_0

BS_1

BS_2

BS_3

Bass Part
Generation
Algorithm

C'

Bass Part D'
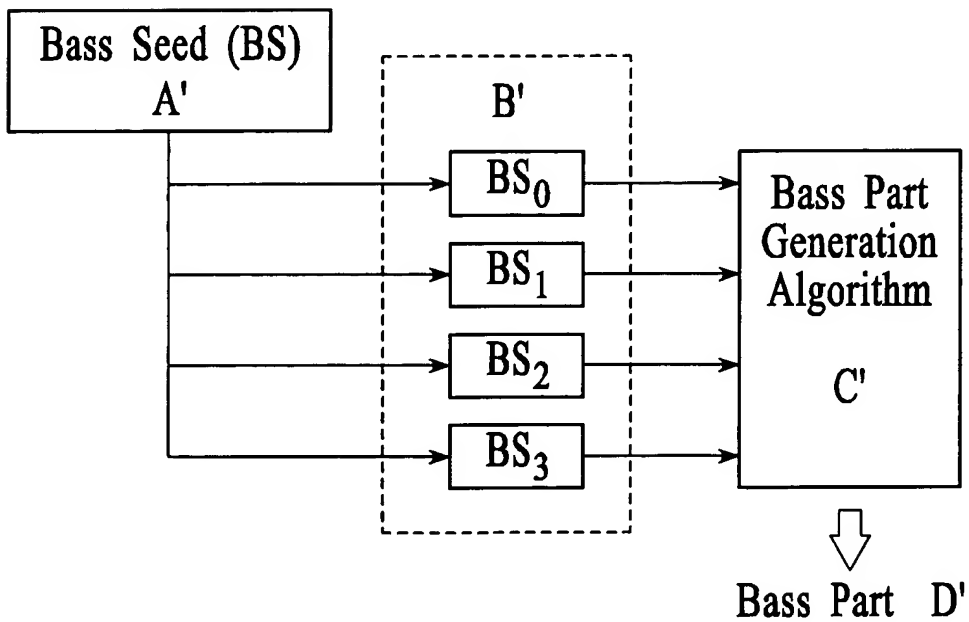
Pseudo-Random Number Implementation 2

FIG. 27

Application Revision	Firmware/application version used to generate the data structure
Style, SubStyle	The style and/or substyle
Sound Bank, Synth Type	The sound bank/synth type
Sample Frequency	How often a sample is played in song
Sample List	List of samples associated with the Style
Key	First Key used, pitch offset
Tempo	Start Tempo (e.g., in pulses per quarter note)
Instrument	Identification of a particular instrument in an instrument group. Indexed by type of instrument
State	State of instrument indexed by instrument type (e.g., muted, un-muted, normal, Forced play, solo, etc.)
Parameter	Instrument parameters indexed by instrument type (e.g., volume, pan, timbre, etc.)
PRNG Seed Values	Seed values used to initialize the PRNG routines

Simple Data Structures

FIG. 28

Start

Determine/Load initial seed values

Generate music with song structure for a given song part

Update user interface

Receive user input? —Yes→ Is user input 'Save'? —Yes→ Save all seeds (as part of Song Structure) to non-temporary Memory Storage

Is user input 'Save'? —No→ Update any relevant seeds in temporary storage

Receive user input? —No→ End of song part?

End of song part? —No→

End of song part? —Yes→ End of song?

End of song? —Yes→ End

End of song? —No→ Go to next song part. Determine/load relevant seeds if necessary.

Example of SDS Flow
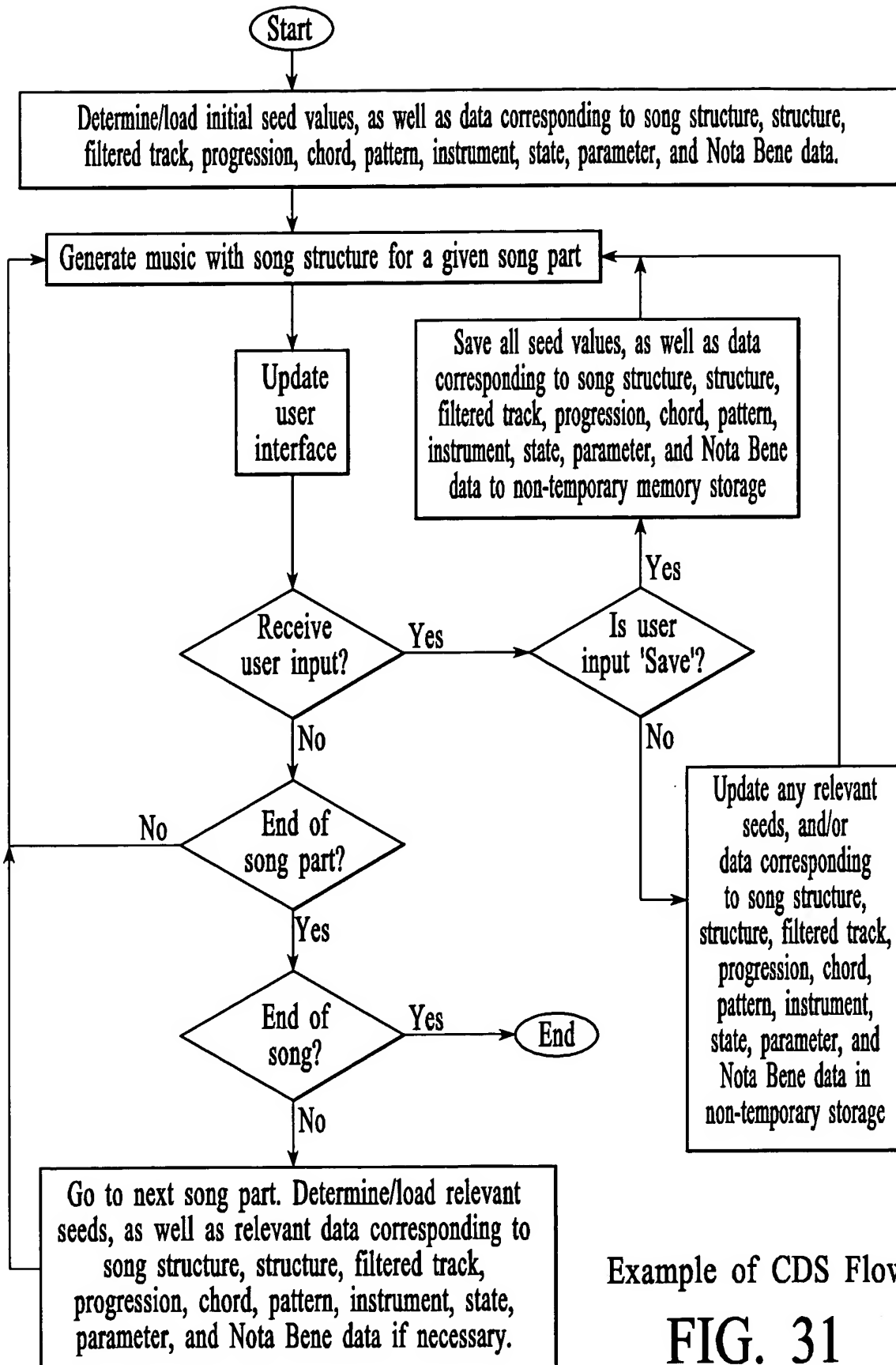
FIG. 29

Application Revision	Firmware/application version used to generate the data structure
Style, SubStyle	The style and/or substyle
Sound Bank, Synth Type	The sound bank/synth type
Sample Frequency	How often a sample is played in song
Sample List	List of samples associated with the Style
Key	First Key used, pitch offset
Tempo	Start Tempo (e.g., in pulses per quarter note)
Song Structure	Number of types, number of parts, sequence of parts, etc.
Structure	For every part: number of sub-parts, sequence of sub-parts, etc. Indexed by Part
Filtered Track	Type, function (e.g., sawtooth wave, sine wave, square wave, etc.), initial value, etc., of an effect. Indexed by Part.
Progression	Time signature, number of SEQs, list of maked types, etc. Indexed by Sub-Part.
Chord	Time stamp, chord vector, key note, progression mode, etc. Indexed by Sub-Part.
Pattern	Combination (Instrument), block data, effects data, etc. Indexed by Type.
Combination	List of instruments. Sub-set of 'Pattern' above.
FX Pattern	Effects data. Sub-set of 'Pattern' above.
Blocks	Block data. Subset of 'Pattern' above.
Instrument	Identification of a particular instrument in an instrument group. Indexed by type of instrument
State	State of instrument indexed by instrument type (e.g., muted, un-muted, normal, Forced play, solo, etc.)
Parameter	Instrument parameters indexed by instrument type (e.g., volume, param1, param2, etc.)
Nota Bene	Improvisation data (e.g., certain instruments or notes) that might be different each time the song is played.

Complex Data Structures

FIG. 30

Start

Determine/load initial seed values, as well as data corresponding to song structure, structure, filtered track, progression, chord, pattern, instrument, state, parameter, and Nota Bene data.

Generate music with song structure for a given song part

Update user interface

Save all seed values, as well as data corresponding to song structure, structure, filtered track, progression, chord, pattern, instrument, state, parameter, and Nota Bene data to non-temporary memory storage

Receive user input? — Yes → Is user input 'Save'?

Yes ↑

No ↓

End of song part? — No

Yes

Update any relevant seeds, and/or data corresponding to song structure, structure, filtered track, progression, chord, pattern, instrument, state, parameter, and Nota Bene data in non-temporary storage

End of song? — Yes → End

No

Go to next song part. Determine/load relevant seeds, as well as relevant data corresponding to song structure, structure, filtered track, progression, chord, pattern, instrument, state, parameter, and Nota Bene data if necessary.

Example of CDS Flow

FIG. 31

SMC
40

Portable Hardware
Device
35

LCD
45

Buffer
59

Keyboard
43

Latch
44

Flash
49

MP
ADD
37

MP
DATA
38

Flash
41

RAM
48

DSP
ADD
46

DSP
DATA
47

MP
36

DSP
42

Hardware
Codec
52

USB
39

Recharge
Battery

+3.3V

3.3V
Regul.

FM
Receiver
50

Connector 53
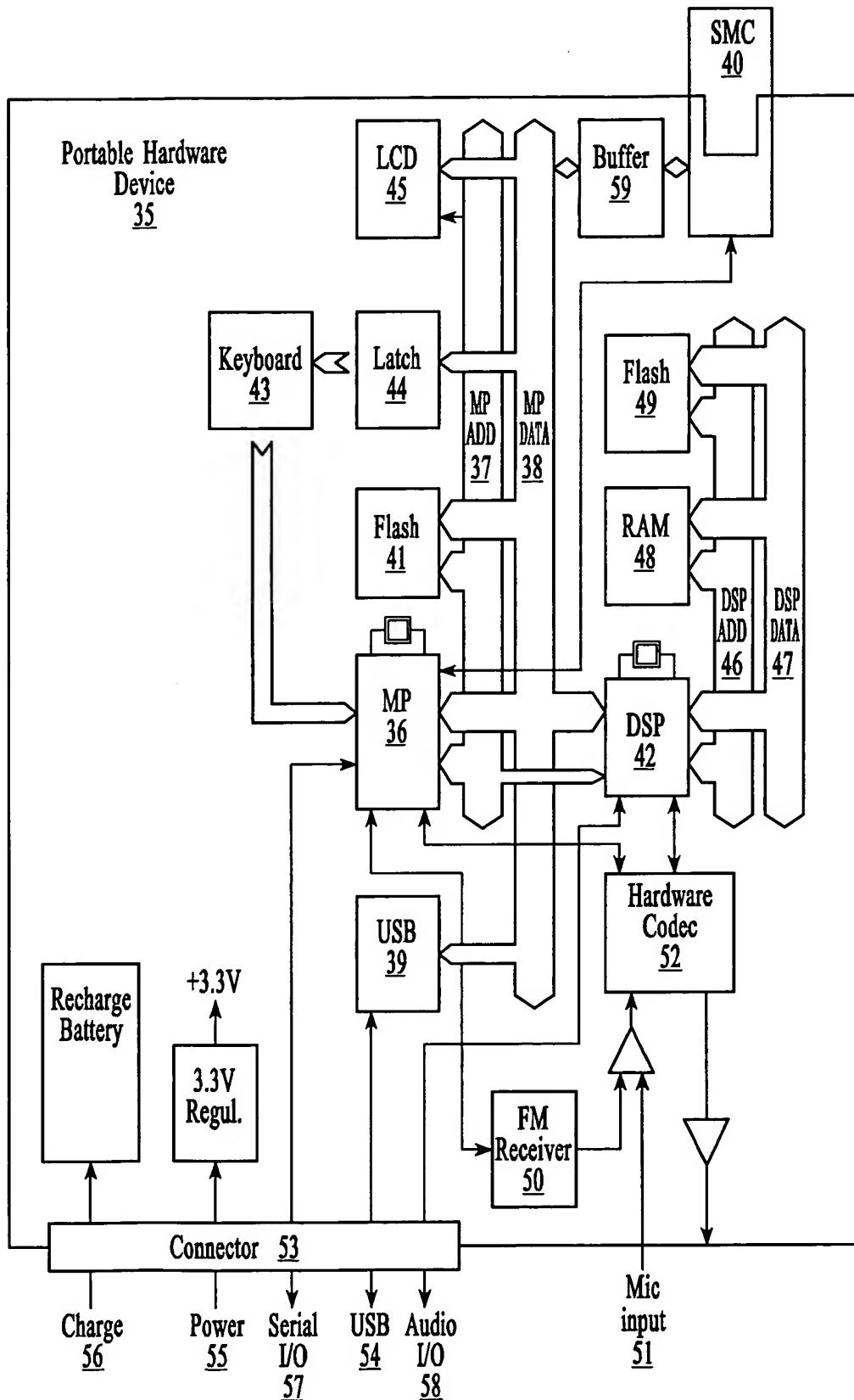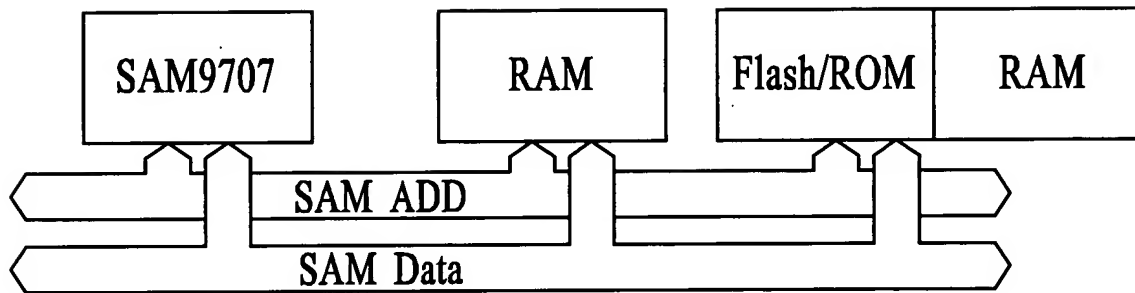
Charge
56

Power
55

Serial
I/O
57

USB
54

Audio
I/O
58

Mic
input
51

FIG. 32

Additional Variation

FIG. 33

DC

Docking Station
Device
70

Power
Supply
79

Flash
76

LMP
ADD
73

LMP
DATA
74

+5V

LMP
71

USB
72

Charge

+3.3V

Connector

Serial

USB

Audio I/O

Audio
Codec
77

Video
I/F
78

Audio
In

Audio
Out

Video
In

Video
Out

USB

MIDI
I/O
75

FIG. 34

0x20000

General Purpose

Split into Subsections

Streaming Buffers

Multi-Use

WMA Code (32 bit)

Block Buffers

Song Buffers

SMC Buffers

Always Loaded In RAM

32bit Routines

32bit Libraries

0x0

Address Map for MP RAM

FIG. 35
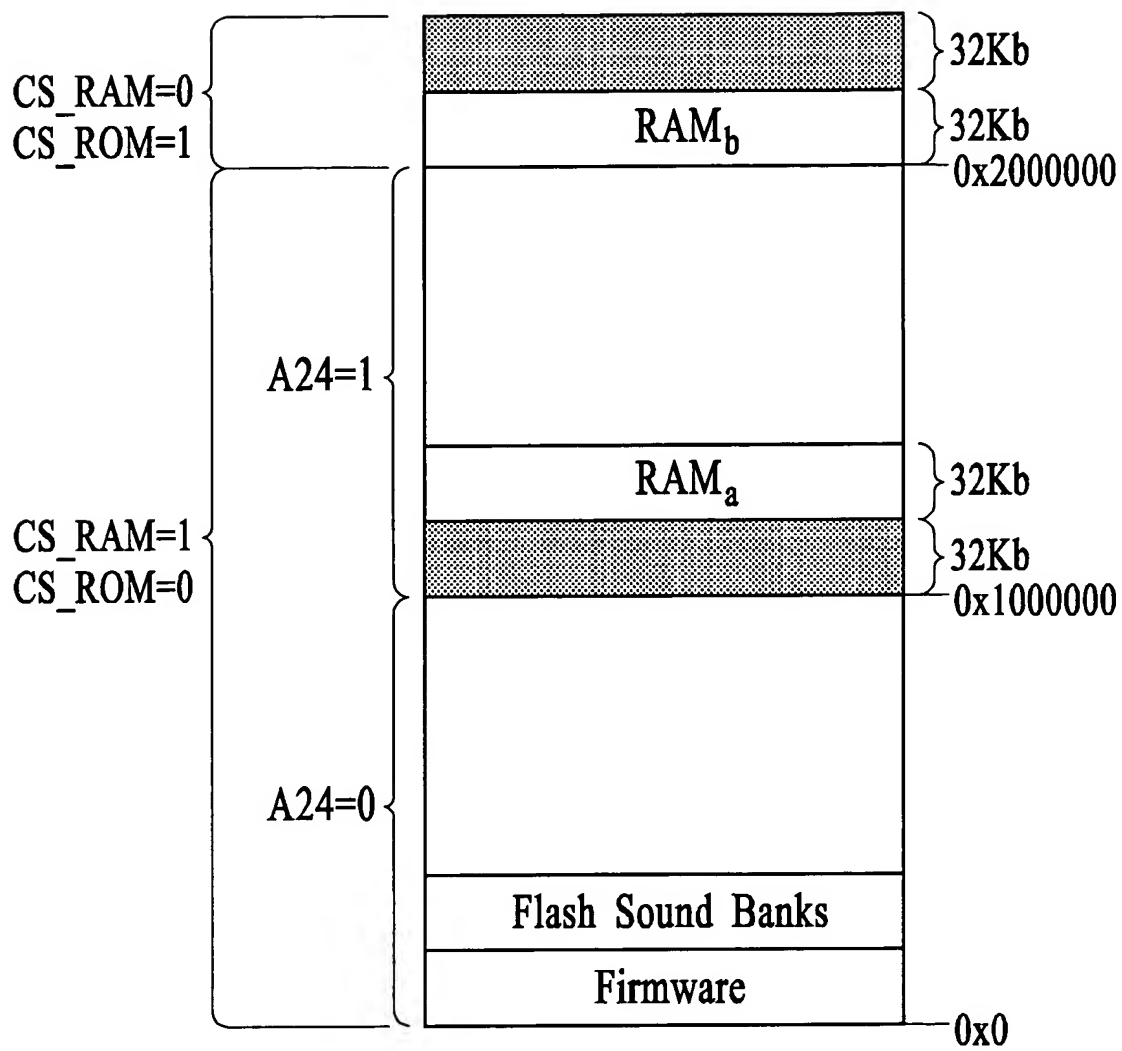
DSP-Local RAM/Flash Address Space

FIG. 36

A24 \ BOOT	0	1
0	Flash	RAM
1	RAM	Flash

Bootstrap Mode Addressing

FIG. 37

CS_RAM		0		1	
A24					
CS_ROM		0	1	0	1
BOOT					
0	0	NA	NA	Flash	RAM
	1	RAM	RAM	NS	NS
1	0	NA	NA	RAM	Flash
	1	NA	NA	NS	NS

Normal Mode

Upgrade Mode

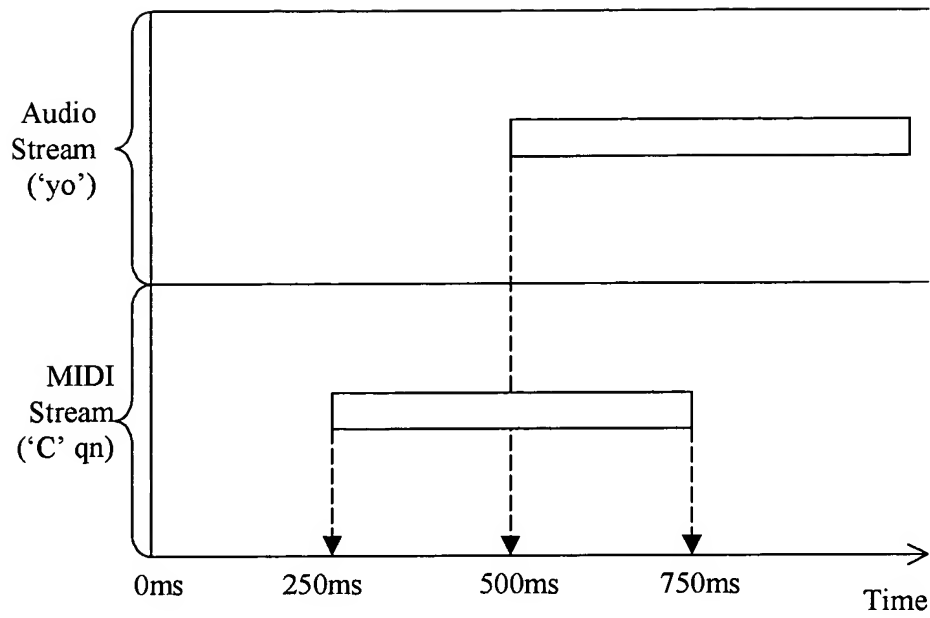CS_RAM and CS_ROM are active low

NS = Nothing Selected

NA = Not Applicable

FIG. 38
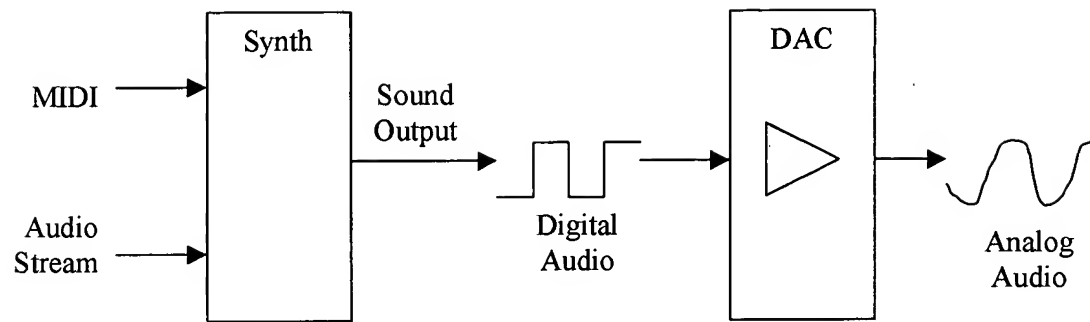
MIDI/Audio Stream

FIG. 39

Simplified MIDI/Audio Stream Timeline

FIG. 40

	NRPN Stream (Hexadecimal)	Indication/Meaning
1	B0	Channel Number
2	63	NRPN Controller A (e.g., audio sample type)
3	40	Identification of sample type (e.g., long, short, stereo, mono, etc.)
4	00	Delta time
5	62	NRPN Controller B (e.g., audio effects type)
6	00	Identification of effects type (ping pong, ripple, phaser, distortion, etc.)
7	00	Delta time
8	06	Identification of register for NRPN Controller A value
9	03	NRPN Controller A value (play 3rd audio sample in set, '00' is random)
10	00	Delta time
11	26	Identification of register for NRPN Controller B value
12	07	NRPN Controller B value (apply audio effect #7, '00' is random)

Simplified NRPN Example

FIG. 41

Simplified Special MIDI Type File

FIG. 42